[AIR-SPIR Module Specification]

Purpose:
- Handles GPU-accelerated vector computation for AIR pipeline using SPIR-V shaders
- Focused on quantization, filter ops, format mapping (float <-> fixed)

Note: This module was co-developed via GPT-4o-based prompt-driven vibe coding in collaboration with 제현.

---

[1] Module Functions
- spir_quantize_subband(input_vec_fp, bits): Return quantized Q-levels from FP input
- spir_vecmap_q31(input_vec_fp): Convert float32 to Q31 int format using normalization shader
- spir_filter(input_vec, kernel): Apply FIR/IIR filter on GPU
- compile_shader(shader_code): Compile SPIR-V shader on runtime and return handle
- execute_shader(handle, input_buf, output_buf): Launch shader with given buffers

---

[2] Shader Types
- Q31_Normalizer.spv: normalize [-1.0 ~ 1.0] float32 to Q31 int32
- Quantizer.spv: subband-wise quantization using dynamic range + bit budget
- FilterBank.spv: polyphase filter bank processing for subband decomposition

---

[3] Integration Points
- AIR-Runtime calls SPIR module at Stage 4 (Quantization)
- Supports batch execution (Wave32 or Wave64)
- Output is Q31 or int8/int16 quantized bins

---

[4] Notes
- Vulkan backend required (API v1.2+ recommended)
- Can be emulated in software fallback if no GPU present (slow)
- Quantization precision affects downstream entropy coding efficiency

Author: 제현
Module: AIR-SPIR
Date: 2025-10-15